Zoea

Research Note 17

The Next AI Language

Edward McDaid & Sarah McDaid
15 Nov 2021

Composable inductive programming is the next stage in the evolution of software development. What does this mean for AI?

The 'AI effect' is a long-standing observation concerning people's attitudes to AI. The idea is that when ever a particular problem - such as playing draughts - has been mastered by an AI system, then people tend to no longer regard that problem as really requiring intelligence - artificial or otherwise. The effect is pithily captured in Larry Tesler's Theorem: AI is whatever hasn't been done yet.

At the end of the day all AI systems are software. This is true regardless of how they are built or trained. What sets AI systems apart from other software is that they are often much more complicated and they are also difficult to map on to conventional sequential computing constructs. In addition the requirements (or domain knowledge) are often very difficult to ascertain. In a sense AI is software we don't really know how to write.

There might seem to be a contradiction between the intractable complexity of AI problems and the use of programming to define the solution. While this might be true for a given problem in its entirety it is still often possible to define elements of the solution as software. The intelligent part is how these elements can be combined to produce solutions.

So does AI need a programming language? Considering recent developments it would be understandable to think that creating AI through programming is mostly a thing of the past. Deep learning has had a lot of success over the last few years and as a result AI is increasingly playing some sort of role in may aspects of modern life. Yet there are many

kinds of problems for which deep learning is not a suitable approach. It is also difficult to provide an explanation of how a particular conclusion was reached. This together with the risk of bias and potentially dystopian use cases could turn many people off AI. Given all this it is increasingly acknowledged that if we want to build AI systems of all kinds and have them explainable then a broader range of approaches will be required. This includes a return to some form of symbolic AI programming.

Whether we need special purpose languages for AI is another question. The short answer is 'yes' because of the distinction between regular software and AI. Conventional programming languages are designed to enable coders to specify the operations that must be carried out and the precise order in which these must happen. However, this sort of rigid approach doesn't work very well when both the operations and their sequencing are uncertain, and can depend on information we don't have when a system is put together. Instead AI systems need to be engineered to be less prescriptive about how a solution is produced.

AI languages tend to adopt declarative approaches where code is structured as rules but the order of execution is driven by the data. This is definitely the right direction but if anything existing AI languages do not go far enough in this respect. While the order in which rules are fired is determined at run time, the composition and sequencing of core language elements within rules is normally fixed. In part this is understandable given that in the past AI languages evolved from conventional imperative languages. Also, early AI developers naturally arose from the ranks of conventional coders.

Encoding knowledge as rules is relatively straightforward but it is often much harder to determine what the rules are in the first place. Historically knowledge elicitation was carried out by people and it is easy to see how all sorts of problems can be introduced at this stage. Knowledge elicitation can be made more effective through the provision of special purpose user interfaces but there is also a strong case for the increased application of intelligence and automation to this part of the process. Indeed, it is salient to recognise that the process used to construct AI systems is an important factor in determining their success.

By this stage anyone who has been following the development of Zoea may start to see some parallels. Zoea is an inductive programming system that allows anyone to

incrementally generate software of any size from a set of test cases. Test cases serve as the basis for a very simple and highly declarative programming language. While Zoea is built using AI it was not explicitly designed to support the development of AI software. Having said that there is also nothing in particular to prevent Zoea from being extended to accommodate this.

Zoea works in part because it contains a large number of knowledge sources that encode expertise about programming language elements and how these can be combined in various ways to form conventional programs. Composable inductive programming is the associated iterative software development process which also plays a crucial role.

In broad terms Zoea users describe the requirements for the software they want as a set of test cases. Zoea uses AI to produce the corresponding software automatically. The resulting software can then be evaluated and if necessary the test cases are extended or adjusted. This process repeats until the resulting software is acceptable.

Conceptually, it is not difficult to map any form of data or model on to a set of test cases. In principle Zoea could be extended with any amount additional knowledge sources that encapsulate the expertise required to build AI systems. This applies regardless of technology and would provide a substrate for integrating disparate approaches such as neural networks and knowledge-based systems.

In the past people coded elements of AI systems by hand and sometimes with mixed results. It is clear there is a place for code in AI - at the very least to serve as an explanation. Given the complexity of AI problems the obvious solution is to reassess the respective roles that people and systems play in the construction of AI systems. What we need is AI systems that produce new AI systems.

Learn more at **zoea.co.uk**